一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
SS2018-22,DC2018-23（2018-10）

組込みシステムの要求仕様書を対象とした状態遷移モデル作成支援

中村　成† 山本　椋太† 吉田　則裕† 高田　広章†

† 名古屋大学大学院情報学研究科　〒464–8603 愛知県名古屋市千種区不老町
E-mail: †{naru,muku,yoshida,hiro}@ertl.jp

あらまし　要求仕様書は多くの場合，自然言語で記述される．そのため，目視による確認では発見できない記述漏れや不整合が存在しうる．これらの確認方法には，状態遷移モデルが用いられる．しかし，膨大な規模の要求仕様書の場合は状態遷移モデル作成時間が増加しうる．そこで，本稿では組込みシステムの要求仕様書から状態遷移モデルの作成を支援するツールを提案し，利用例を示す．

キーワード　自然言語処理，組込みシステム，要求仕様書，状態遷移モデル

Supporting the Generation of State Transition Models in a Requirements Specification Document for an Embedded System

Naru NAKAMURA†, Ryota YAMAMOTO†, Norihiro YOSHIDA†, and Hiroaki TAKADA†

† Graduate School of Infomatics, Nagoya University,　Furo-cho, Chikusa-ku, Nagoya, Aichi, 464–8603 Japan
E-mail: †{naru,muku,yoshida,hiro}@ertl.jp

Abstract　Generally, it is difficult for developers to find all insufficient or inconsistent descriptions in a requirement specifications document during a review process especially in the case of using a natural language. State transition model (STM) can be used during the review of a requirement specifications document. However, it takes much time to generate STMs in a large-scale requirement specifications document. In this paper, we propose a tool for generating STMs in a requirement specifications document for an embedded system, and then show a usage example of the proposed tool.

Key words　Natural Language Processing, Embedded System, Requirements Specification Document, State Transition Model

1. はじめに

システム開発における要求仕様書は多くの場合，自然言語で記述される．そのため，目視による確認のみでは発見できない記述漏れや不整合が存在しうる [1] [2]．また，組込みシステムの大規模化・複雑化により開発期間の短縮と品質向上が求められている [3]．要求仕様書の品質と開発の成否は大きく関わっており，開発を成功させるためにも，要求仕様書中の誤りを早い段階で検出し，除去することが望まれる [4]．そのため，要求定義において，記述漏れや不整合を確認できる状態遷移モデルが用いられる [5]．しかし，膨大な規模の要求仕様書から状態遷移モデルを手作業で作成する場合，開発者は要求仕様書から状態遷移モデル作成に必要な記述を目視で見つける必要があるため，多大な時間を要する [6]．

そこで本稿では，組込みシステムの要求仕様書から状態遷移モデルの作成を支援するツールを提案する．この作成支援を実現するために，提案ツールは状態遷移モデル作成に必要な状態遷移記述候補を抽出する．本稿における状態遷移記述は，状態に関わる記述，条件に関わる記述およびシステムの動作に関わる記述である．これらの記述は，状態に関わる記述である状態名と状態値，条件に関わる記述である状態値判定とイベント，システムの動作に関わる記述である遷移と処理から成る．

本研究では提案ツールとして，2 つの機能を実装した．機能 1 は，ユーザが提案ツールに要求仕様書を与えることで，各文に存在する節の抽出・分類を行う．この機能は，我々の既存研究 [6] における成果（以下，節の抽出・分類ツールと呼ぶ）を利用する．本稿における節は状態遷移記述候補を含む要素である．抽出した節は，定義節，条件節，処理節のいずれかに分類される．我々の既存研究 [6] においては，節の抽出・分類ツールに改良の余地があるため，本研究では，このツールの改良および機能拡張を行う．機能 2 は機能 1 で抽出・分類した節を用いて，状態遷移記述候補を抽出する．機能 1 は自動だが，機能 2 は半自動である．機能 2 では，状態名候補，状態値候補，状態値判定候補，イベント候補，遷移候補・処理候補の順に抽出

表 1　分類する節に関する情報

節分類	状態遷移記述候補	節の分類条件
定義節	状態名候補	文中に条件節が存在しない
	状態値候補	
条件節	状態値判定候補	節の末尾の品詞列が 以下のいずれかである ・副詞的名詞 + 格助詞　（例：時に） ・副詞的名詞 + 接続助詞（例：時の） ・副詞的名詞 + 副助詞　（例：時は） ・副詞的名詞 + 読点　（例：時、）
	イベント候補	
処理節	遷移候補	直前の節が条件節である
	処理候補	

する．このうち，状態名候補・状態値候補・状態値判定候補・イベント候補は，提案ツールが候補となりうる記述を表示し，ユーザが候補を選択する．遷移候補・処理候補はユーザが選択した内容から提案ツールが自動抽出する．

本稿では，実務者のレビューによる修正が加えられた要求仕様書に対して，提案ツールを適用する．その後，筆頭著者が要求仕様書を目視で参照して手作業で状態遷移表を作成する場合と，筆頭著者が提案ツールを用いて抽出した状態遷移記述候補を参照して手作業で状態遷移表を作成する場合を比較して，提案ツールの評価を行う．

2. 節の抽出・分類ツール

我々の既存研究 [6] で提案されている節の抽出・分類ツールは 2 つの処理を行う．まず，ユーザが節の抽出・分類ツールに要求仕様 1 文を与えることで，このツールは状態遷移記述候補を容易に抽出することを目的とした独自の木（以下，解析木と呼ぶ）を導出する．解析木の導出手順は以下の（**手順 1**）から（**手順 8**）である．

（**手順 1**）　半角文字を全角文字に変換し，単位記号の表記を日本語に変換する（例：ms ＝ ミリ秒）．

（**手順 2**）　形態素解析システム JUMAN(注1)による形態素解析 [7] を行う．

（**手順 3**）　未定義語と分類された形態素の品詞を名詞に置換する．

（**手順 4**）　隣り合う名詞を結合し，複合名詞を作る．

（**手順 5**）　形態素列に対して我々が定義した規則（例：副詞的名詞 + 読点）を適用して，1 文を節単位に分割する．

（**手順 6**）　日本語構文・格・照応解析システム KNP(注2) [8] による構文解析 [7] を行う．

（**手順 7**）　KNP の出力結果である基本句，文節，係り受け情報 [8] を用いて部分木のノード情報とエッジ情報を作成する．

（**手順 8**）　部分木のノード情報とエッジ情報を組み替えることで，句読点と節の結合および修飾語と名詞の結合を含む解析木を導出する．

次に，解析木から 1 文に存在する節を抽出し，考案した分類条件を適用することで，節を分類する．節分類，節に対応する状態遷移記述候補および節の分類条件を表 1 にまとめる．

節の抽出・分類ツールは，1 文で完結する文に対して，我々の手作業による分類と結果が一致している [6]．しかし，箇条書きは，誤った分類になる．

（注1）：http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN

（注2）：http://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP

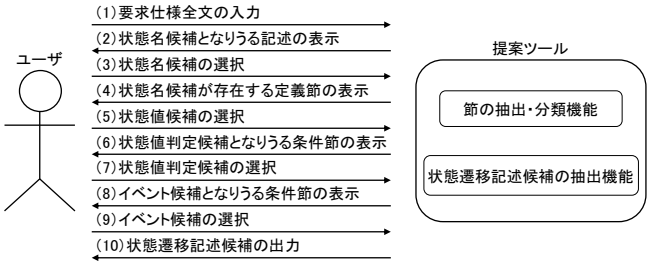

図 1　提案ツール利用の流れ

表 2　節単位に分割するために追加する規則

規則
指示詞（代名詞）以外 + 副詞的名詞 + 副助詞 + 読点 （例：以下の条件が成立している場合は、）
指示詞（代名詞）以外 + 副詞的名詞 + 格助詞 + 読点 （例：撮影ボタンが OFF の時に、）
名詞 + 動詞 + 読点 （例：撮影ボタンを ON できる条件が成立していることを判定し、）

3. 提案ツール

提案ツールの流れを図 1 に示す．図に示すように，（1）から（10）の手順で，状態遷移記述候補を出力する．提案ツールは，節の抽出・分類機能と状態遷移記述候補の抽出機能を組み合わせたものである．節の抽出・分類機能には，2. で説明した節の抽出・分類ツール [6] を利用する．しかし，節の抽出・分類ツールには改良の余地があるため，本研究では，このツールの改良および機能拡張を行う．

本章では，節の抽出・分類ツールの改良および機能拡張と状態遷移記述候補の抽出機能について説明する．

3.1 節の抽出・分類機能の改良および機能拡張

3.1.1 解析木の導出における機能拡張

ユーザは節の抽出・分類ツール [6] に要求仕様 1 文を与えるが，2. で述べたように，箇条書きの分類誤りが起こる．そこで，ユーザが提案ツールに与える入力を要求仕様全文にする．

節の抽出・分類ツール [6] は形態素解析の前処理に 2. の（**手順 1**）を行う．しかし，節の抽出・分類ツールが行う処理は要求仕様 1 文単位であるため，要求仕様全文を 1 文単位に分割する処理を（**手順 1**）の前に追加する．

節の抽出・分類ツール [6] は構文解析の前処理に 2. の（**手順 3**）から（**手順 5**）を行う．しかし，（**手順 5**）において，分割する規則が不十分であったため，表 2 の規則を追加する．また，文頭に副詞 + 読点（例：または、）もしくは接続詞 + 読点（例：しかし、）が存在する場合は，解析上不必要であるため，それらを除去する処理を追加する．

表 3　本研究で用いる属性情報

属性	ラベル	例
type	action（動作）	表示する
	state（状態）	ON である
	epithet（形容詞）	多い
	others（その他）	画面を
parallel	and（連言）	速度かつ
	or（選言）	速度または
attribute	interrogative（疑問詞）	何を
	condition（条件）	時に
	restriction（制限）	したので

3.1.2　条件節分類の改良

節の抽出・分類ツール [6] は表 1 の分類条件を用いて節を分類する．しかし，品詞およびその細分類と見出し語の組み合わせは多岐に渡るため，実装コストが大きい．そこで，KNP が出力する feature 情報 [8] を利用した文節への属性付与 [9] を用いる．feature 情報とは，形態素や文節の働きを示す情報（例：<自立> [10]）である．feature 情報を活用して，文節の属性分けを提案している研究がある [9]．本研究では [9] で示されている属性のうち，type，parallel，attribute を用いる．これらの属性が持つラベルを表 3 にまとめる．

関係語とは，type 属性が action，state または， 文末の epithet である文節である．attribute 属性が condition または，restriction かつ関係語である文節を終端文節として，終端文節までに存在する形態素列を条件節とする．また，1 文に parallel 属性（and/or）を含む文節があれば，その文節を含む節と，その節の直前に存在する節を結合する．ここで（例 1）を考える．

（例 1）

モード切替ボタンを『オン』にすると、モード選択画面を表示する。

節の抽出・分類ツール [6] は（例 1）の句点までの 1 文を定義節に分類する．しかし，本来は条件節「モード切替ボタンを『オン』にすると、」と処理節「モード選択画面を表示する。」に分類することが望ましい．そこで，表 3 の属性を用いて分類すると，期待通りの条件節と処理節に分類可能であった．表 3 の属性は，（例 1）のような表 1 の分類条件では分類できない条件節を分類でき，加えて我々の既存研究 [6] で分類できたものをすべて分類できることも確認できている．

3.1.3　箇条書きの分類

要求仕様書に記述されている箇条書きの文章構造は<空白>$^{+}$<文>である．ここで，“+” は 1 回以上の繰り返しを表す．本稿における箇条書きは，その箇条書きよりも前の<空白>の数が 1 段少ない文に存在する「以下の」が含まれている節を詳細に説明していることとする．箇条書きの判定には，文頭に存在する<空白>の個数（ネストの深さ）を用いる．節の抽出・分類ツール [6] が箇条書きを除く要求仕様全文に対して節の分類を行った後，（**a**）と（**b**）を行い，箇条書きを分類する．

（**a**）　箇条書きを参照する節を探索し，抽出する．

（**b**）　（**a**）で得た節の分類に合わせて箇条書きを分類し，（**a**）で得た節と紐付ける．

（**a**）で探索する節は「以下の」が含まれており，箇条書きよりもネストの深さが 1 浅いものとする．ここで，（例 2）を考える．

（例 2）

以下の全ての条件が成立している場合、スイッチを『オン』にする。
・スイッチ押下判定が『オン』である。
・入力が許可である。

（例 2）では，箇条書きを参照する節「以下の全ての条件が成立している場合、」が条件節であるため，「スイッチ押下判定が『オン』である。」と「入力が許可である。」は “箇条書きの条件” に分類され，「以下の全ての条件が成立している場合、」と各箇条書きが紐付く．

3.2　状態遷移記述候補の抽出機能

抽出する状態遷移記述候補の定義を以下に示す．

- 状態名候補：ユーザ選択した，システムの状態を表す名詞
- 状態値候補：ユーザ選択した，状態名候補がとりうる値
- 状態値判定候補：状態名候補と状態値候補を含む条件節，またはどちらか一方のみを含む条件節のうちユーザ選択された条件節
- イベント候補：状態値判定候補に伴う条件節，または状態値判定候補を除く条件節のうちユーザ選択された条件節
- 遷移候補：状態値判定候補とイベント候補の両方もしくはどちらか一方を伴う，状態名候補を含む処理節
- 処理候補：状態値判定候補とイベント候補の両方もしくはどちらか一方を伴う，状態名候補を含まない処理節

このうち，状態名候補，状態値候補，状態値判定候補，イベント候補は，抽出する際に対象システムに関する知識が必要であるため，ユーザが選択する．

3.2.1　状態名候補と状態値候補の抽出

状態名候補の抽出を行うために，ユーザに状態名候補となりうる記述を表示して，ユーザが状態名候補を 1 つ選択する機能を実現する．抽出手順は以下の（**抽出 1-1**）と（**抽出 1-2**）からなる．

（**抽出 1-1**）　定義節に存在する主語を抽出し，その主語を状態名候補となりうる記述とする．

（**抽出 1-2**）　ユーザに状態名候補となりうる記述を表示し，ユーザがそれを目視で確認して，状態名候補を 1 つ選択する．本稿で対象とする要求仕様書には，1 文字の名詞が状態名候補となる場合が存在しないため，状態名候補となりうる記述から除去する（例：表）．

状態値候補の抽出を行うために，状態名候補が含まれている定義節を表示して，ユーザが状態値候補を選択する機能を以下の（**抽出 2**）により実現する．

（**抽出 2**）　ユーザに状態名候補が含まれている定義節を表示し，ユーザがそれを目視で確認して，状態値候補を選択する．定義節が箇条書きを参照する節の場合，紐付く箇条書きもユー

ザに表示する．状態値候補の抽出に定義節を用いる理由は，条件節だと，定義節で定義されている状態値候補を全て用いていない場合が考えられるためである．

3.2.2 状態値判定候補の抽出

状態値判定候補の抽出を行うために，ユーザに状態値判定候補となりうる条件節をランキング形式で表示して，ユーザが状態値判定候補を選択する機能を実現する．ランキング作成には，2 つの文字列の類似度を測るレーベンシュタイン距離 [11] を用いる．このランキングは，3.2.1 で表示した定義節と関連の強い条件節が上位に現れるランキングである．抽出手順は以下の（**抽出 3-1**）から（**抽出 3-6**）である．

（**抽出 3-1**） 1 文に存在する条件節を抽出する（以下，対象条件節と呼ぶ）．

（**抽出 3-2**） 対象条件節と 3.2.1 で表示した定義節のレーベンシュタイン距離を算出する．

（**抽出 3-3**） 対象条件節が feature 情報の parallel 属性（and/or）を含む場合，1 つに結合する．結合後，結合前の対象条件節のレーベンシュタイン距離のうち最小値を保持する．

（**抽出 3-4**） 対象条件節に紐付く箇条書きを 1 つに結合する．このとき，箇条書きがネスト構造になっているならば，再帰的に結合する．結合後，結合前の箇条書きのレーベンシュタイン距離のうち最小値を保持する．

（**抽出 3-5**） 算出したレーベンシュタイン距離に基づいて，対象条件節を昇順に並び替えることで，以下のランキングを作成する．

- 状態名候補と状態値候補を表す文字列が存在するランキング
- 状態名候補を表す文字列が存在するランキング
- 状態値候補を表す文字列が存在するランキング

（**抽出 3-6**） ユーザは各ランキングを目視で確認して，状態値判定候補となる条件節を選択する．

3.2.3 イベント候補の抽出

イベント候補は 2 種類に細分することができる．1 つ目は状態値判定候補を含む文に存在する条件節である．そのため，この条件節は，イベント候補として確定する．イベント候補の抽出は以下の（**抽出 4-1**）により，実現する．

（**抽出 4-1**） 状態値判定候補を含む文に存在する状態値判定候補以外条件節を抽出し，その条件節がイベント候補となる．また，その条件節に紐付く箇条書きもイベント候補となる．

2 つ目は状態値判定候補を伴わない文に存在する条件節である．そのため，この条件節は，イベント候補として確定できない．以降は，イベント候補として確定できない条件節を不確定なイベント候補と呼ぶ．不確定なイベント候補においては，ユーザに不確定なイベント候補をランキング形式で表示して，ユーザがイベント候補となる条件節を選択する機能を実現する．ランキング作成には，3.2.2 と同様にレーベンシュタイン距離 [11] を用いる．このランキングは，状態値判定候補と関連性の強い条件節が上位に現れるランキングである．抽出手順は（**抽出 4-2-1**）から（**抽出 4-2-6**）である．

（**抽出 4-2-1**） 不確定なイベント候補を抽出する．

（**抽出 4-2-2**） 不確定なイベント候補と状態値判定候補のレーベンシュタイン距離を算出する．

（**抽出 4-2-3**） 不確定なイベント候補が feature 情報の parallel 属性（and/or）を含む場合，1 つに結合する．結合後，結合前の条件節のレーベンシュタイン距離のうち最小値を保持する．

（**抽出 4-2-4**） 不確定なイベント候補に紐付く箇条書きを 1 つに結合する．このとき，箇条書きがネスト構造になっているならば，再帰的に結合する．結合後，結合前の箇条書きのレーベンシュタイン距離のうち最小値を保持する．

（**抽出 4-2-5**） 算出したレーベンシュタイン距離に基づいて，不確定なイベント候補を昇順に並び替えることで，ランキングを作成する．

（**抽出 4-2-6**） ユーザは各ランキングを目視で確認して，イベント候補となる条件節を選択する．

3.2.4 遷移候補・処理候補の抽出

遷移候補・処理候補の抽出を行うために，以下の並びで存在する処理節を自動抽出する機能が必要である．

- <状態値判定候補> + <イベント候補> + <処理節>
- <イベント候補> + <処理節>

抽出手順は（**抽出 5-1**）から（**抽出 5-3**）である．

（**抽出 5-1**） 状態値判定候補を含む文に存在する処理節と，状態値判定候補を含まないが，イベント候補を含む文に存在する処理節を探索し，その処理節を抽出する（以下，対象処理節と呼ぶ）．

（**抽出 5-2**） 対象処理節に紐付く箇条書きを抽出する．このとき，箇条書きがネスト構造になっているならば，再帰的に探索する．

（**抽出 5-3**） 対象処理節もしくは箇条書きに，状態名が存在すれば遷移候補，状態名が存在しなければ処理候補となる．

4. 使 用 例

本章では，筆頭著者が要求仕様書を目視で参照して手作業で状態遷移表を作成する場合と，筆頭著者が提案ツールを用いて抽出した状態遷移記述候補を参照して手作業で状態遷移表を作成する場合を比較する．本事例の対象は，企業において実際に使用されている車載システムの要求仕様書である．この要求仕様書は，ある大学院生のレビュー結果に基づいて改善されており，129 項目から成る．

4.1 使用手順と結果

提案ツールは CUI で実装している．筆頭著者が提案ツールに対象の要求仕様全文を与えて，状態遷移記述候補を抽出する．状態名候補となりうる記述のうち，本事例では提案ツールを実装する上で，筆頭著者が要求仕様書を目視で確認した際に，状態値判定が多く存在する印象を受けたシステム ON/OFF 状態を状態名候補に選択した．

状態名候補にシステム ON/OFF 状態を選択した場合の状態遷移記述候補を表 4 にまとめる．表 4 に記述されている各行の状態値判定候補，イベント候補，遷移候補・処理候補はそれぞれ紐付いている．状態遷移表の作成手順は，まず横軸に表 4 の

表 4　システム ON/OFF 状態を状態名に選択した場合の状態遷移記述候補

状態名候補	状態値候補	状態値判定候補	イベント候補	遷移候補・処理候補
システム ON/OFF 状態	「オン」「オフ」	システム ON/OFF 状態が「オン」である時に	以下の全ての条件が成立している場合、 ・ON/OFF スイッチ押下判定が「オン」である。 ・メイン ON が許可である。	システム ON/OFF 状態を「オン」から「オフ」にする。（遷移候補）
		システム ON/OFF 状態が「オフ」である時に	以下の全ての条件が成立している場合、 ・ON/OFF スイッチ押下判定が「オン」である。 ・メイン ON が許可である。	システム ON/OFF 状態を「オフ」から「オン」にする。（遷移候補）
		システム ON/OFF 状態が「オン」である時に	以下の条件が成立している場合、 ・メイン ON が許可でない。	システム ON/OFF 状態を「オン」から「オフ」にする。（遷移候補）
		なし	システム ON/OFF 状態が「オン」から「オフ」に遷移した場合、	以下の情報に対して初期値を設定する。 （処理候補） ・システムモード ・車間設定 ・目標速度
		なし	以下のいずれかの条件が成立している場合、 ・システムモードが「定速」モードである。 ・システム ON/OFF 状態が「オフ」である。	車間設定を初期値にする。（処理候補）
		なし	以下の全ての条件が成立している場合、 ・システム ON/OFF 状態が「オン」である。 ・システムセット状態が「セット中」でない。	システムモードを切り替えられる条件が成立していると判定する。（処理候補）

表 5　筆頭著者が提案ツールを用いて手作業で作成した状態遷移表

イベント候補＼状態候補	システムON/OFF状態 『オン』	システムON/OFF状態 『オフ』
以下の全ての条件が成立している場合、 ・ON/OFFスイッチ押下判定が『オン』である。 ・メインONが許可である。	ー システムON/OFF状態を『オン』から『オフ』にする。	ー システムON/OFF状態を『オフ』から『オン』にする。
以下の条件が成立している場合、 ・メインONが許可でない。	ー システムON/OFF状態を『オン』から『オフ』にする。	ー ー
システムON/OFF状態が『オン』から『オフ』に遷移した場合、	以下の情報に対して初期値を設定する。 ・システムモード ・車間設定 ・目標速度 ー	ー ー
以下のいずれかの条件が成立している場合、 ・システムモードが『定速モード』である。 ・システムON/OFF状態が『オフ』である。	車間設定を初期値にする。 ー	車間設定を初期値にする。 ー
以下の全ての条件が成立している場合、 ・システムON/OFF状態が『オン』である。 ・システムセット状態が『セット中』でない。	システムモードを切り替えられる条件が成立していると判定する。 ー	ー ー

表 6　筆頭著者が提案ツールを用いず，手作業で作成した状態遷移表

イベント候補＼状態候補	システムON/OFF状態 『オン』	システムON/OFF状態 『オフ』
以下の全ての条件が成立している場合、 ・ON/OFFスイッチ押下判定が『オン』である。 ・メインONが許可である。	以下の情報に対して初期値を設定する。 ・システムモード ・車間設定 ・目標速度 システムON/OFF状態を『オン』から『オフ』にする。	車間設定を初期値にする。 システムON/OFF状態を『オフ』から『オン』にする。
以下の条件が成立している場合、 ・メインONが許可でない。	以下の情報に対して初期値を設定する。 ・システムモード ・車間設定 ・目標速度 システムON/OFF状態を『オン』から『オフ』にする。	車間設定を初期値にする。 ー
システムモードが『定速モード』である。	車間設定を初期値にする。 ー	車間設定を初期値にする。 ー
システムセット状態が『セット中』でない。	システムモードを切り替えられる条件が成立していると判定する。 ー	車間設定を初期値にする。 ー

状態名候補と状態値候補を記述する．その際に状態名候補と状態値候補をまとめて，状態候補とする．次に縦軸に表 4 のイベントを記述する．最後に，表 4 の状態値判定候補と隣り合うイベント候補の組で起こる遷移候補・処理候補を，状態値判定候補に存在する状態値候補と対応するイベント候補が交差するセルに記述する．また，状態値判定候補がない場合は，イベントに存在する状態値候補を見て，その状態値候補とイベント候補が交差するセルに遷移候補・処理候補を記述する．セルの内側の四角に遷移候補，四角の外側に処理候補を書く．表 4 を参照して，筆頭著者が手作業で作成した状態遷移表を表 5 に示す．また，筆頭著者が提案ツールを用いず，手作業で作成した状態遷移表を表 6 に示す．表 4 のイベントの 1 行目と 2 行目は文字列完全一致する記述であるため，表 5 と表 6 を作成する際には，それらを統合した．作成時間は，表 5 が約 30 分，表 6 が約 50 分である．

4.2 考　　察

表 5 と表 6 を比較すると，一致するセルと異なるセルの存在を確認できる．以下では，相違点について説明し，考察する．

表 6 が理想的な状態遷移表であるにも関わらず，遷移候補に空のセルが存在する．これが起こる原因は 2 つ考えられる．1 つ目は，要求仕様書の抜け・漏れである．2 つ目は，要求仕様書に書かれていなくても問題がない場合である．しかし，提案ツールはどちらが原因であっても，それを指摘できない．必ず自状態遷移する場合ならば，遷移候補を書くことができるが，そもそも遷移候補を記述する必要がない場合も存在し得る．そのため，必要がない遷移候補はユーザが手作業で削除する，もしくは提案ツールが必要のない遷移候補であることを示す実装が考えられる．

表 6 では，システム ON/OFF 状態が『オン』である時の 1 つ目のセルと 2 つ目のセルに処理候補と遷移候補が同時に存在するが，表 5 では，遷移候補のみ存在する．前提として，提案ツールは要求仕様書に書かれていない記述を抽出することができない．そのため，抽出できない記述は要求仕様書に存在しない場合があると考えられる．本事例では，提案ツールが抽出したイベント候補の意味を筆頭著者が解釈することで，表 5 で埋めることができなかった処理候補・遷移候補を表 6 の通り埋めた．

表 5 の 3 行目のイベント「システム ON/OFF 状態が『オン』から『オフ』に遷移した場合、」が表 6 に存在しない．これは，提案ツールの UI でイベント候補と選択した「システム ON/OFF 状態が『オン』から『オフ』に遷移した場合、」に対応する処理候補が，状態遷移する瞬間に生じる処理候補となるためである．表 6 の 1 つ目と 2 つ目のセルの存在する処理候補は，それぞれのセル内に記述されている遷移候補が生じた後，その遷移候補に基づいて実行される処理候補である．表 5 では対応できていない．したがって，提案ツールの改良が必要である．これを実現するには，表 4 中で遷移候補に基づいて実行される処理候補を表す列を新たに設けるか，ユーザが遷移候補に基づいて実行される処理候補であることを判定することを考えている．

表 5 の 4 行目と 5 行目のイベント候補の表記が表 6 と異なる．これは，表 5 のイベントの 4 行目の箇条書き「システム ON/OFF 状態が『オフ』である。」と 5 行目の箇条書き「システム ON/OFF 状態が『オン』である。」が状態値判定候補となるためである．表 5 の 4 行目のイベントは，OR 関係の箇条書きであるため，2 つの箇条書きのうち少なくとも一方が成立すれば，処理候補「車間設定を初期値にする。」が行われる．そのため，表 6 の 3 行目のイベント候補から箇条書きの状態値判定が除去され，『オフ』の列に「車間設定を初期値にする。」を記述する．表 5 の 5 行目のイベント候補は，AND 関係の箇条書きであるため，2 つの箇条書きが成立すれば，処理候補「システムモードを切り替えられる条件が成立していると判定する。」が記述される．そのため，表 6 の 4 行目のイベント候補から箇条書きの状態値判定が除去され，「システム ON/OFF 状態が『オン』である。」かつ「システムセット状態が『セット中』でない。」に対応するセルに「システムモードを切り替えられる条件が成立していると判定する。」を記述する．これを提案ツールで実現するには，状態遷移記述候補の抽出前に，箇条書きを参照する節を feature 情報の parallel 属性（OR/AND）で結合した箇条書きに置換し，状態遷移記述候補の抽出を行えば良いため，実装が容易だと考えられる．

表 5 と表 6 の作成時間を見ると，提案ツールを用いた場合は，完全手作業と比べて 4 割程度の時間短縮を実現することができた．今後，提案ツールの UI を CUI から GUI に変更する予定であるため，更なる時間短縮が見込める．

5. おわりに

本稿では，組込みシステムの要求仕様書から状態遷移モデルの作成を支援するツールを提案し，その節の抽出・分類機能の改良および機能拡張と状態遷移記述候補の抽出機能について説明した．次いで，使用例の結果，状態遷移記述候補は，ある程度抽出可能であることが確認でき，提案ツールは完全手作業の場合と比べて 4 割程度の時間短縮を実現することができた．

今後の課題は，遷移後に行われる処理候補の分類と，箇条書きの状態値判定候補とそれが含む feature 情報の parallel 属性（OR/AND）を抽出する処理を実現するために，提案ツールを改良し，最終的な被験者実験を行うことである．

謝辞　本研究におけるデータ収集に協力していただいた，名古屋大学の清水 貴裕氏に感謝致します．

文　　献

[1] 山本椋太，吉田則裕，高田広章，“組込みシステムの要求仕様書に対する修正候補の定量的調査，” 電子情報通信学会技術研究報告，vol.117，no.249，pp.49–54，2017.
[2] 位野木万里，近藤公久，“省略と修飾パターンを用いた用語不一致検証による要求仕様の一貫性検証支援ツールの実現と適用評価，” コンピュータソフトウェア，vol.35，no.3，pp.109–127，2018.
[3] 高田広章，“組込みシステム開発技術の現状と展望，” 情報処理学会論文誌，vol.42，no.4，pp.930–938，2001.
[4] 山田さつき，大森隆行，大西淳，“要求文書からの信頼性要求の抽出と検証，” 情報処理学会研究報告，vol.2018-SE-199，no.28，pp.1–8，2018.
[5] 渡辺政彦，“状態遷移ベースのソフトウェア開発環境の現状と動向，” 計測と制御，vol.41，no.2，pp.117–121，2002.
[6] 中村成，山本椋太，吉田則裕，高田広章，“組込みシステムを対象とした要求仕様書からの状態遷移記述の抽出，” 情報処理学会研究報告，vol.2018-SE-198，no.5，pp.1–8，2018.
[7] 黒橋禎夫，自然言語処理，放送大学教育振興会，2015.
[8] 黒橋禎夫，“日本語構文解析システム KNP version 4.1 使用説明書，” http://nlp.ist.i.kyoto-u.ac.jp/?KNP，2013.
[9] 村上響一，村上神龍，久代紀之，牧茂，田畑一政，神代勉，中村潤，“自然言語仕様書からの試験ケース生成のための条件・動作の同定手法，” 情報処理学会研究報告，vol.2018-SE-198，no.7，pp.1–7，2018.
[10] 河原大輔，“KNP で付与される feature 一覧，” http://nlp.ist.i.kyoto-u.ac.jp/?KNP，2013.
[11] V.I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” Soviet physics doklady, vol.10, no.8, pp.707–710, 1966.